**Renz\_01**

1. Which kind of traversal generates what kind of expression?
2. How to insert parenthesis correctly into infix expressions?
3. Implement the generation of a fully parenthesized infix expression!

Exercise – Lab Preparation:

1. Consider the example in slides 13/16.
2. Write down the postfix expresssion of this tree.
3. Execute the Stack Algorithm for the evaluation of this postfix expression.
4. Find out by what means you can – instead of calculating – construct the arithmetic tree

(Hint: Stack<BiNode>)

Exercise – Exam Preparation:

1. Determine the postfix expression that corresponds to the mathematical expression



1. Determine the maximum operand stack size required when evaluating that postfix expression
2. Draw the corresponding arithmetic tree.
3. Write the fully parenthesized infix expression

**Renz\_02**

Nothing seems to be valuable =)

**Renz\_03**

Determine the time complexity of algorithm A!

What’s to be done?

1. Identify the most time consuming operations of A
2. Count the num of operations (Formula)
3. Determine the asymptotics (N -> infinity) of the num of operations
4. Complexity is the maximum order of all asymptotics.

Space complexity addresses memory needed.

Exercise for exam:

1. Count the num of swaps and comparisons for the following input

“it”,“was”,“the”,“best”,“of”,“times”,“ever”,“seen”

using Insertion sort and Selection sort

1. Compare your findings with the theoretical bounds known

**Renz\_04**

1. Count the num of swaps and comparisons for the following input

“it”,“was”,“the”,“best”,“of”,“times”,“ever”,“seen”

using Insertion sort and Selection sort

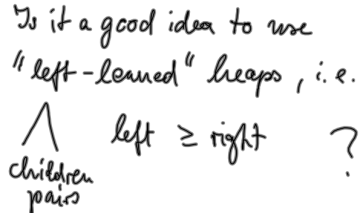
+ using Mergesort and Quicksort

1. Compare your findings with theoretical bounds known (formulas)

**Renz\_06**

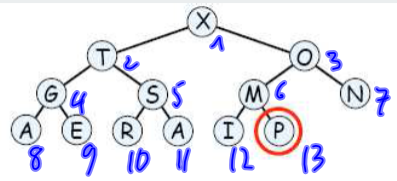
Exercise problem:

Is it a good idea to use “left-leaned” heaps, i.e. left >= right?



Exercise:

Count nums of less (?) and nums of exchanges



1. For restoring the heap after removing the “x” in the above example
2. For sorting “HEAPSORTING” with heap sort

(look at heapsort building in Renz’s lecture slides)

**Renz\_07**

(look at putroot at the end of Renz’s lecture slides #7)

**Renz\_08**

Exercise:

Now it is not the worst of times ever seen

1 2 3 4 5 ….. 10

1. Insert the sequence of words into a BST and count num of comparisons (words) needed for each put-command

\*(?) as key, value is the sequence number (I)

value is the num of comparisons (II)

1. When inserting the word, into a RandomBST:
2. What is the probability to call “putroot” in each put call?
3. Construct the random BST for the case where “putroot” is called only for the put(“seen”) and count num of comparisons(word) in this call

Some answers can be seen in Renz’s slides #8

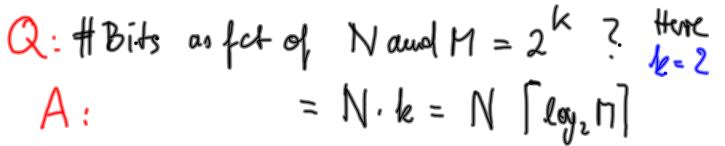
**Renz\_09**

Question:

Num of bits of N and M = 2^k?

Answer:

N\*k=N\*…… (look in the picture)

 notice it’s not [ ] !!!!!

Question:

Consider the information (content) that comes (to the receiver with the appearance of) a character

Answer:

That info depends on frequency of char -> the more frequent, the less info

Exercise:

Assume a random sequence of elements of alphabet

